REMARKS

Claims 16, 38, 41, and 43 have been amended, claim 42 has been canceled, and claims 44 – 46 have been added. Claims 1 – 34 have been allowed. Hence, 1 – 34 and 38 – 41, and 43 – 46 are pending in the application.

Summary of Rejections

Claims 38-40 were rejected under 35 U.S.C. 103(a) as being unpatentable over (US 5,907,847), herein Goldberg in view of (US 6,374,252), hereinafter Althoff.

Claim 41 was rejected under 35 U.S.C. 103(a) as being unpatentable over Althoff in view of (US 5,600,005), hereinafter Hoover.

Claims 42-43 were rejected under 35 U.S.C. 103(a) as being unpatentable over Althoff and Hoover in view of Goldberg.

Claims 1 – 24 and 26 – 35 have been allowed and renumbered as claims 1 – 34. These claims are not further discussed. These claims are presented herein as re-numbered.

Claim 38

Claim 38, recites:

A method performed by one or more computers, comprising:
storing data in a table of a database that is managed by a database server;
wherein the table is defined by a table definition;
wherein an object class is defined by an object class definition;
maintaining, separate from the table definition and object class definition, metadata that indicates how to derive object ids from values stored in the table; and
the database server deriving, based on the metadata, object ids for objects of the object class from the values in the table.

Claim 38 requires a "database server deriving, based on the metadata, object ids for objects of the object class from the values in the table", where the metadata "indicates how to

derive object ids from values stored in the table." These features are not taught or suggested by the prior art.

To establish prima facie obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art. (MPEP § 2143.03, citing In Re Royka, 490 F.2d 981, 180 USPQ 580 (CCPA 1974)) The cited references, however, do not teach or suggest all the claim limitations of claim 38.

The Office Action alleges Althoff teaches "the database server deriving object ids for the data in the table based on the metadata." This allegation is incorrect. Althoff, as well as Goldberg, fail to teach this feature of claim 38.

Althoff, describes a method and system for modeling object-oriented database structures, translation to relational database structures, and performing searches thereon. (Abstract). Althoff does discuss object ids. The following is the only excerpt in Althoff related to creating object ids.

> The system 200 creates, for each table 800, a column 810 "Object ID" for the UID of the object, and assures that objects added to each table 800 for derived classes 101 are also added to tables 800 for the base classes 101, with corresponding UIDs. For example, each row 820 added to the table 800 static memory corresponds to a row 820 added to the table 800 memory and corresponds to a row 820 added to the table 800 component, and the UIDs are identical in the column 810 "Object ID" for these three rows 820. (col. 24, lines 9 – 17)

The above excerpt discloses how the rows of different tables that represent the same object are assured of having the same value for the object id. It does not follow from the fact that rows of different tables that represent the same object are assured of having the same value for the object id, that object ids are derived based on metadata that indicates how to derive the object id. Thus, the excerpt, nor anything else in Althoff, suggests in any way

much less discloses deriving object ids based on metadata that indicates how to derive an object id.

The Office Action alleges that col. 7, lines 36 – 45, lines 47 – 67, col. 12, lines 59 – 62 disclose "a database server deriving object ids for the data in the tables based on the metadata." (Section 9) The Office Action in particular emphasized col. 7, lines 47 – 67, stating "it is interpreted that Althoff teaches how metadata is used by a database server to derive object ids for data in table..."

These allegations are incorrect. The cited excerpts are provided below.

The user interface 210 receives commands and descriptions from the user 201 regarding the user's object database 100, and in response thereto, builds and edits the user database model 230. As the user database model 230 comprises a set of objects in the meta-model 220, the process of building and editing the user database model 230 simply comprises building and editing objects in the meta-model 220. These objects are maintained by the system 200 as persistent objects, so the user database model 230 is maintained as a persistent part of the meta-model 220 by the system 200.

The meta-model 220 itself is represented by a meta-model relational database 221, comprising a set of relational database tables, properties of those tables, columns within those tables, primary and secondary keys of those tables, and other defining features of the meta-model relational database 221. The user database model 230 is represented by objects in the meta-model relational database 221, comprising rows within those tables, specific values entered in the columns for those rows, and pointers from a row of one table to a row of a related table. The process of building and editing the user database model 230, i.e., building and editing objects in the meta-model 220, simply comprises building and editing classes 101, objects, searchable properties 102 and relationships 103 between classes 101, represented by rows, values and pointers in the meta-model relational database 221 which represent the descriptive information about the user's object database 100. (col. 7, lines 45 – 67)

When the user 201 creates a derived class 101, or alters the base/derived status of a class 101, the system 200 creates or edits an object of class Class Link 302 to model the base/derived relationship (col. 12 lines 59-63).

The excerpts above describe how a meta-model of a database is represented by objects in a relational database, and how the objects are created and modified in response to user input. Though the term "meta-model" is mentioned, object ids or deriving object ids based on a meta-model or metadata is not described or suggested anywhere in the above excerpts. Thus, the cited sections of Althoff could not possibly teach or disclose "deriving object ids...based on the metadata", as claimed.

While the Office Action states that it interprets at least some of these excerpts to disclose "how metadata is used by a database server to derive object ids for data in table", the Office Action fails to specify any specific item or action in the excerpts that correlates to an object id, that correlates to metadata indicating how to derive an object id, or that correlates to any way of deriving object ids. In an Office Action "the particular part relied on must be designated as nearly as practicable ... The pertinence of each reference, if not apparent, must be clearly explained ..." (37 C.F.R. § 1.104; MPEP 707). As shown above, the pertinence of the excerpts are not apparent and are not clearly explained. Instead, large portions of the references are simply identified in a non-specific way. The failure to specify any item or action in the excerpts that correlates to an object id, metadata indicating how to derive an object id, or deriving object ids, is tantamount to admitting that Althoff fails to describe what the Office Action alleges them to describe.

Like Althoff, Goldberg fails to teach a "database server deriving, based on the metadata, object ids for objects of the object class from the values in the table", where the metadata "indicates how to derive object ids from values stored in the table", as claimed. The Office Action also correlates an object id in claim 38 to the object attribute Employee ID

(Office Action, section 13, Issue No. 1). Even if an object attribute like Employee ID can be correlated in this way, which it cannot, Goldberg nevertheless fails to disclose maintaining metadata that indicates how to derive any value for a particular object attribute much less an object id.

Goldberg describes a system for storing class definitions, behavior, and object state in the same source, i.e. a single database. (Abstract, col. 6, lines 1 – 25). One table (or a relation of multiple tables) holds class definitions and behaviors for objects. (i.e. table 404, col. 6, lines 41 – 45) Another table holds data for the objects (employee table 412, col. 6, lines 53-55). While Goldberg discloses details about class definitions being stored in a table and object state being stored in a table, Goldberg teaches nothing about metadata that indicates how to derive an individual attribute or object id.

The following excerpts in Goldberg are all Applicant found that related to how object attributes, like Employee ID, are derived.

> When a request is received, processing continues step 512 to process the object request. At step 512, the object's state is retrieved from the DBMS' data relation(s). At step 514 (i.e., "have behavior?"), a determination is made whether the behavior for that class of object has already been retrieved. If not, processing continues at step 516 wherein the object behavior and class definition are retrieved from the schema relation(s). After the retrieval operation is performed at step 516, or if it is determined that the object's behavior has already been retrieved, processing continues at step 518. The new object instance having both state and behavior is provided to the requester, or client. The object is thereby activated in the client environment. For example, the state and behavior are copied into executable address space (e.g., a client process) to allow execution of the object's behavior on the object's state. (col. 8, lines 50 – 65)

> When the DBMS server determines that the emp object is to be retrieved as a result of the client request, for example, processing continues at step 512. At step 512, the object state (e.g., for the emp object) is retrieved from the data relations stored at the

DBMS server. As discussed above, an object's behavior could have previously been retrieved from the DBMS. If not, however, the object's behavior is retrieved at step 516. At step 518, the object (with its state and behavior) is activated in the DBMS server environment (e.g., copied to executable address space in the DBMS server environment). At step 520, the method, or behavior, (e.g., getManager) is invoked in the DBMS server environment. (col. 9, lines 28 – 40)

Previously, at least two separate and distinct steps were required to retrieve an object's state and behavior. In a library environment such as that used in existing object-oriented environments, the state is retrieved from the DBMS and the behavior is retrieved from a file system.

As discussed above, an object's state and behavior can be retrieved from one store (e.g., the DBMS server) using the present invention. State and behavior can therefore be retrieved in a single step, or transaction. Retrieval can be performed using one or more select statements in a single transaction, for example. That is, a single transaction can be used to retrieve the state and the behavior stored in the DBMS server. A transaction is a mechanism used in a DBMS environment whereby a sequence of database operations can be packaged together and performed as a unit. (col. 9, lines 53 – 62)

As shown above, Goldberg teaches how state for objects are retrieved from a table using a transaction. None of the excerpts describe or suggest in any way how an individual attribute is derived much less how an object id is derived. In fact, the Office Action states Goldberg fails to teach the database server deriving object ids for the data in the tables based on the metadata.

Based on the foregoing, it would not have been obvious to one having ordinary skill in the art at the time the invention was made to modify Goldberg in view of Althoff's teachings because the cited art, alone and in combination, fails to teach all the limitations of claim 38. Reconsideration and allowance of claim 38 is requested.

Based on the foregoing, claim 38 is patentable.

Claim 41

Claim 41 recites:

A method performed by a database server, the method comprising the steps of:
the database server executing a query statement that conforms to a query language
and that references an object view as if the object view were a table;
wherein metadata for a database:
defines the object view as a presentation of data as a set of objects in the database, and
indicates that data for the object view is in one or more tables of the database, the one or more tables including at least one relational table; and
wherein executing the query statement includes:
reading data from one or more rows of the one or more tables indicated by the metadata that defines said object view, and
presenting said data from the one or more rows as objects that reside in said database.

As amended, claim 41 requires "executing a query statement that conforms to a query language and that references an object view as if the object view were a table," where metadata defines the object view "as a presentation of data as a set of objects in the database." Such a feature is not disclosed or suggested in any way by the cited art.

Claim 41 has been rejected as being unpatentable over Althoff in view of Hoover. Althoff describes a method and system for modeling object-oriented database structures, translation to relational database structures, and performing searches thereon. (Abstract). Hoover describes a system and methods for transforming data stored in a plurality of remote, heterogeneous, object-relational databases into a homogeneous data model (Abstract).

The Office Action has rejected claim 41 based on the allegation that Althoff teaches "executing a query that references an object view as if the object view were a table." Specifically, the Office Action cites col. 29 lines 3-67 and col. 30 lines 1-52 of Althoff as

teaching "the method for presenting data from a set of one or more tables in a database, the method comprising the steps of: in response to executing a query that references an object view as if the object view were a table." There is no basis for this allegation. Because there is no basis for this allegation, there is no basis for rejecting claim 41, which, as amended, requires "executing a query statement that conforms to a query language and that references an object view as if the object view were a table."

Col. 29, lines 3-67 and col. 30, lines 1-52 describe a portion of "a process for cascade searching of an object database and search translation between object-oriented and relational database structures" (col. 24, lines 34-36). Specifically, at "step 936, the system 200 examines the query model 260, the list of query model objects 992, the alias records 993, the join records 994, and the condition records 995, and in response, generates SQL commands 261 using the form shown in table 9-3." (col. 29, , lines 3 – 6)

Again, the Office Action fails to specify what items and actions in the cited excerpts have been correlated to what features in claim 41. Large portions of the references are simply identified in a non-specific way. The pertinence of the excerpts are not apparent and are not clearly explained. As a result, Applicant has had to engage in guesswork about what has been correlated.

Perhaps the Office Action is correlating the query that references an object view as if the object view were a table to the SQL commands generated in Althoff. Althoff does describe how a section of the generated SQL commands reference tables. With regard to the section of the generated SQL commands that reference tables, Althoff states the following: "Information regarding the tables (and aliases of tables) in the relational database 250 to select data from is inserted in the <tables and aliases> section of the SQL commands 261."

(col. 29, lines 38 – 40). This excerpt simply teaches that the generated SQL command references the table as the table or an alias for the table. An alias is simply another label for a table. This excerpt, however, contains nothing about referencing a view as a table, much less a query statement that references an object view as a table, as claimed. If the generated SQL commands cannot suggest in anyway a query statement that reference an object view as a table, it cannot possibly disclose or suggest a query statement that both "references an object view as if the object view were a table", and defines "a presentation of data as a set of objects in the database," as claimed.

Perhaps the Office Action is correlating a query that references an object view as if the object view were a table to the query model generated in Althoff. Note claim 41, as amended, requires not just a query but a "query statement that conforms to a query language" The query model cannot be correlated to the query statement as claimed. The following describes how a query model is built.

> At a step 920, the system 200 parses the cascade search request and builds the query model 260. To perform this step 920, the system 200 performs the steps 921 through 922 inclusive.
>
> At a step 921, the system 200 builds a configuration object 991 for the initial class 101 selected by the user 201 to be searched. Each such configuration object 991 represents a single class 101 selected by the user 201, and comprises a first list of searchable properties 102 to be searched, a second list of searchable properties 102 to be displayed in the query result 251, and a third list of classes 101 starting with the initial class 101 selected by the user 201, and continuing with each related class 101 selected by the user 201.
>
> At a step 922, the system 200 builds a configuration object 991 for each additional class 101 selected by the user 201 to be searched, and links each such additional

configuration object 991 to the configuration object 991 for the initial class 101 selected by the user 201 to be searched.

As shown above, the query model is a list of objects. Nothing about a list of objects discloses or suggests a statement that conforms to a computer language, much less a "query statement that conforms to a query language". If the query model does not suggest in anyway a "query statement that conforms to a query language", it cannot possibly disclose or suggest a "query statement that conforms to a query language", that "references an object view as if the object view were a table", and that defines "a presentation of data as a set of objects in the database."

Based on the foregoing, none of teachings of Althoff disclose or suggest in anyway a "a query statement that conforms to a query language and that references an object view as if the object view were a table" and that defines "a presentation of data as a set of objects in the database." This feature is also not suggested in any way by Hoover. In fact, the Office Action only alleged that Althoff discloses "a query that references an object view as if the object view were a table", but has not alleged that Hoover does so. Therefore, the cited art does not teach or suggest all the claim limitations of claim 41, and claim 41 is patentable. Reconsideration and allowance of claim 41 is requested.

Unallowed Pending Claims

The unallowed pending claims not discussed so far are dependant claims that depend on an independent claim that is discussed above. Because each of the dependant claims include the limitations of claims upon which they depend, the dependant claims are patentable for at least those reasons the claims upon which the dependant claims depend are patentable. Removal of the rejections with respect to the dependant claims and allowance of the dependant claims is respectfully requested. In addition, the dependent claims introduce

additional limitations that independently render them patentable. Due to the fundamental difference already identified, a separate discussion of those limitations is not included at this time.

It is respectfully requested that the Examiner reconsider all of the pending claims, which are now in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

The Examiner is respectfully requested to contact the undersigned by telephone if it is believed that such contact would further the examination of the present application.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Dated: December 22, 2004

Marcel K. Bingham
Reg. No. 42,327

1600 Willow Street
San Jose, CA 95125
Telephone No.: (408) 414-1080 ext.206
Facsimile No.: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Mail Stop RCE, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

on 12/22/04 by Jennifer Newell